NCR

NCR Corporation
Dayton, Ohio 45479

World Headquarters

EP-9823-01 0579

NCR

NCR IMOS COBOL

Pocket Reference

THIRD EDITION

NOTATION IN GENERAL FORMATS

Notation	Indicates
Underlined, uppercase word	Key word that must be present and must be spelled correctly
Uppercase word	Optional word that is not really needed; however it must be spelled correctly if used
Lowercase word	Operand representing user-defined word, literal, PICTURE character-string, or comment-entry; this operand must be present.
Brackets [ ]	Enclosed portion of general format is included or omitted at the programmer's option; stacking shows choices to be made by programmer
Braces { }	Enclosed portion of general format must be present; stacking indicates programmer must select one
Ellipsis ...	Portion enclosed in brackets or braces may be repeated by programmer
Comma or semicolon	These two characters are optional; they are also interchangeable
Period	This character must be present

NCR IMOS COBOL DATA DEFINITION

Data Type	Number of Bits Per Character	PICTURE Characters		USAGE Clause	SIGN Clause
		S	Data Characters		
Alphanumeric	8	No	X	DISPLAY or unspecified	No
Unsigned decimal	8	No	9	DISPLAY or unspecified	No
Trailing signed decimal	8	Yes	9	DISPLAY or unspecified	TRAILING SEPARATE
Trailing zone-signed decimal	8	Yes	9	DISPLAY or unspecified	TRAILING or unspecified
Unsigned packed decimal	4	No	9	COMP 6 (or COMP)	No
Signed packed decimal	4	Yes	9	COMP 3 (or COMP)	No
Unsigned binary	..	No	9	COMP 5	No
Boolean	1	No	1	BIT	No

FILE STATUS VALUES

Content of File Status Data Item	Status of Input-Output Operation	OPEN	CLOSE	READ	WRITE	REWRITE	DELETE	START
00	Successful completion of input-output operation	SRI	SRI	SRI	SRI	SRI	RI	RI
10	At end condition			SRI				
21	Invalid key due to sequence error				I	I		
22	Invalid key due to duplicate key				RI			
23	Invalid key due to no record found			RI		RI	RI	RI
24	Invalid key due to boundary violation				RI			
30	Permanent error condition due to:							
	• hardware read error	SRI	SRI	SRI				
	• hardware write error		SRI		SRI	SRI	RI	
	• physical block length error			SRI				
34	Permanent error condition due to boundary violation				S			
90	Illegal open operation due to file not available	SRI						
91	Illegal open operation due to file not closed	SRI						
92	Illegal operation due to file not opened		SRI	SRI	SRI	SRI	RI	RI
93	Illegal operation for declared open mode			SRI	SRI	SRI	RI	RI
94	Illegal operation due to boundary violation			SRI				
95	Illegal open operation due to wrong attribute	SRI						
96	Illegal open operation due to device mismatch	SRI						
97	Invalid magnetic tape label	S						
98	Physical record length different from record length declared in COBOL source program			SRI	SRI	SRI		
99	Sector not available on shared disc file			SRI	SRI	SRI	RI	RI
	Blank magnetic tape			S				
9A	Illegal open operation due to insufficient memory	SRI						
9B	Illegal operation due to file organization mismatch	S						
9C	Condition of last I-O operation on card reader printer, or magnetic tape file not known due to previous power failure	S	S	S	S	S		
9D	Clear trailer encountered on magnetic tape			S	S			
0E	Tape mark on magnetic tape			S				
0F	End of tape hole encountered on magnetic tape			S	S			
	Boundary violation warning on disc file				SRI			

S = sequential file organization
R = relative file organization
I = indexed file organization

PROGRAMMER SUPPLIED WORDS

Word	Number of Characters	Permissible Characters	Restriction
alphabet-name	1 to 30	A-Z, 0-9, —	At least one alphabetic character; hyphen must be embedded; must be unique
condition-name	1 to 30	A-Z, 0-9, —	At least one alphabetic character; hyphen must be embedded; must be unique
data-name	1 to 30	A-Z, 0-9, —	At least one alphabetic character; hyphen must be embedded; may be qualified by file-name, record-name, or another data-name
file-id	1 to 10	A-Z, 0-9, —	First character must be alphabetic
file-name	1 to 30	A-Z, 0-9, —	At least one alphabetic character; hyphen must be embedded; must be unique
index-name	1 to 30	A-Z, 0-9, —	At least one alphabetic character; hyphen must be embedded; must be unique
level-number	1 or 2	00-49, 77	Need not be unique
mnemonic-name	1 to 30	A-Z, 0-9, —	At least one alphabetic character; hyphen must be embedded; must be unique
paragraph-name	1 to 30	A-Z, 0-9, —	May be all numeric; hyphen must be embedded; may be qualified by section-name
physical-file-id	1 to 10	A-Z, 0-9, —	First character must be alphabetic
record-name	1 to 30	A-Z, 0-9, —	At least one alphabetic character; hyphen must be embedded; may be qualified by file-name
section-name	1 to 30	A-Z, 0-9, —	At least one alphabetic character; hyphen must be embedded; must be unique
segment-number	1 or 2	00-99	Need not be unique
text-name	1 to 30	A-Z, 0-9, —	At least one alphabetic character; hyphen must be embedded; first 10 characters must be unqiue

GENERAL FORMAT FOR IDENTIFICATION DIVISION

```
IDENTIFICATION DIVISION.
PROGRAM-ID. program-name.
[AUTHOR. [comment-entry] ... ]
[INSTALLATION. [comment-entry] ... ]
[DATE-WRITTEN. [comment-entry] ... ]
[DATE-COMPILED. [comment-entry] ... ]
[SECURITY. [comment-entry] ... ]
```

GENERAL FORMAT FOR ENVIRONMENT DIVISION

```
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. NCR-IMOS [WITH DEBUGGING MODE].
OBJECT-COMPUTER. NCR-IMOS
    [, MEMORY SIZE integer { WORDS      }]
                           { CHARACTERS }
                           { MODULES    }
    [, PROGRAM COLLATING SEQUENCE IS alphabet-name].
[SPECIAL-NAMES.
    [, SWITCH-n {ON STATUS IS condition-name-1  [, OFF STATUS IS condition-name-2]}]...
                {OFF STATUS IS condition-name-2  [, ON STATUS IS condition-name-1]}
    [, LINE-n IS mnemonic-name] ...
    [, alphabet-name IS STANDARD-1] ...
    [, CURRENCY SIGN IS literal]
    [, DECIMAL-POINT IS COMMA].]
[INPUT-OUTPUT SECTION.
 FILE-CONTROL.
    {file-control-entry} ...
[I-O-CONTROL.
    [; SAME [RECORD] AREA FOR file-name-1 {, file-name-2} ... ] ... .]]
```


ANSI HOLLERITH PUNCHED CARD CODE

Character	Card Punches	Character	Card Punches	Character	Card Punches	Character	Card Punches
NUL	12-0-1-8-9	¢		@	4-8	`	1-8
SOH	12-1-9	!	12-7-8	A	12-1	a	12-0-1
STX	12-2-9	"	7-8	B	12-2	b	12-0-2
ETX	12-3-9	#	3-8	C	12-3	c	12-0-3
EOT	7-9	$	11-3-8	D	12-4	d	12-0-4
ENQ	0-5-8-9	%	0-4-8	E	12-5	e	12-0-5
ACK	0-6-8-9	&	12	F	12-6	f	12-0-6
BEL	0-7-8-9	'	5-8	G	12-7	g	12-0-7
BS	11-6-9	(	12-5-8	H	12-8	h	12-0-8
HT	12-5-9	)	11-5-8	I	12-9	i	12-0-9
LF	0-5-9	*	11-4-8	J	11-1	j	12-11-1
VT	12-3-8-9	+	12-6-8	K	11-2	k	12-11-2
FF	12-4-8-9	,	0-3-8	L	11-3	l	12-11-3
CR	12-5-8-9	-	11	M	11-4	m	12-11-4
SO	12-6-8-9	.	12-3-8	N	11-5	n	12-11-5
SI	12-7-8-9	/	0-1	O	11-6	o	12-11-6
DLE	12-11-1-8-9	0	0	P	11-7	p	12-11-7
DC1	11-1-9	1	1	Q	11-8	q	12-11-8
DC2	11-2-9	2	2	R	11-9	r	12-11-9
DC3	11-3-9	3	3	S	0-2	s	11-0-2
DC4	4-8-9	4	4	T	0-3	t	11-0-3
NAK	5-8-9	5	5	U	0-4	u	11-0-4
SYN	2-9	6	6	V	0-5	v	11-0-5
ETB	0-6-9	7	7	W	0-6	w	11-0-6
CAN	11-8-9	8	8	X	0-7	x	11-0-7
EM	11-1-8-9	9	9	Y	0-8	y	11-0-8
SUB	7-8-9	:	2-8	Z	0-9	z	11-0-9
ESC	0-7-9	;	11-6-8	[	12-2-8	{	12-0
FS	11-4-8-9	<	12-4-8	\	0-2-8	¦	12-11
GS	11-5-8-9	=	6-8	]	11-2-8	}	11-0
RS	11-6-8-9	>	0-6-8	^	11-7-8	~	11-0-1
US	11-7-8-9	?	0-7-8	_	0-5-8	DEL	12-7-9

A SET AND H SET PUNCHED CARD CODES

Character	A Set & H Set Card Punches
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	12,1
B	12,2
C	12,3
D	12,4
E	12,5
F	12,6
G	12,7
H	12,8
I	12,9
J	11,1
K	11,2
L	11,3
M	11,4
N	11,5
O	11,6
P	11,7
Q	11,8
R	11,9
S	0,2
T	0,3
U	0,4
V	0,5
W	0,6
X	0,7
Y	0,8
Z	0,9

Character	Card Punches	
	A Set	H Set
¢	blank	blank
!	12,7,8	11,0
"	7,8	7,8
# or £	3,8	12,7,8
$	11,3,8	11,3,8
%	0,4,8	0,5,8
&	12	2,8
'(apostrophe)	5,8	4,8
(	12,5,8	0,4,8
)	11,5,8	12,4,8
*	11,4,8	11,4,8
+	12,6,8	12
,(comma)	0,3,8	0,3,8
–	11	11
.(period)	12,3,8	12,3,8
/	0,1	0,1
:	2,8	5,8
;	11,6,8	11,6,8
<	12,4,8	12,6,8
=	6,8	3,8
>	0,6,8	6,8
?	0,7,8	12,0
@	4,8	0,7,8
[	12,2,8	12,5,8
\	0,2,8	0,6,8
]	11,2,8	11,5,8
↑ or ^	11,7,8	11,7,8
← or _	0,5,8	0,2,8

GENERAL FORMAT FOR FILE CONTROL ENTRY

SEQUENTIAL FILE:

```
SELECT file-name

      ASSIGN TO { DISC
                { CASSETTE
                { MAGNETIC-TAPE
                { CARD-READER
                { PRINTER

    [; ORGANIZATION IS SEQUENTIAL]

    [; ACCESS MODE IS SEQUENTIAL]

    [; FILE STATUS IS data-name-1].
```

RELATIVE FILE:

```
SELECT file-name

      ASSIGN TO DISC

    ; ORGANIZATION IS RELATIVE

    [; ACCESS MODE IS { SEQUENTIAL  [, RELATIVE KEY IS data-name-1] }
                      { { RANDOM  }                                 }
                      { { DYNAMIC } , RELATIVE KEY IS data-name-1   }]

    [; FILE STATUS IS data-name-2].
```

INDEXED FILE:

```
SELECT file-name

      ASSIGN TO DISC

    ; ORGANIZATION IS INDEXED

    [; ACCESS MODE IS { SEQUENTIAL }]
                      { RANDOM     }
                      { DYNAMIC    }

    ; RECORD KEY IS data-name-1

    [; FILE STATUS IS data-name-2].
```

GENERAL FORMAT FOR DATA DIVISION

```
DATA DIVISION.

[FILE SECTION.

[FD  file-name

    [; BLOCK CONTAINS  [integer-1 TO]  integer-2  {RECORDS    }]
                                                  {CHARACTERS}

    [; RECORD  {CONTAINS  [integer-3 TO]  integer-4 CHARACTERS                                  }]
               {IS VARYING IN SIZE  [FROM integer-3]  [TO integer-4]  CHARACTERS               }
               {    [DEPENDING ON data-name-1]                                                 }

     ; LABEL  {RECORD IS }  {STANDARD}
              {RECORDS ARE}  {OMITTED }

    [; VALUE OF FILE-ID IS literal-1]

    [; DATA  {RECORD IS }  data-name-1  [, data-name-2] ...]
             {RECORDS ARE}

    [; LINAGE IS  {data-name-4}  LINES  [, WITH FOOTING AT  {data-name-5}]
                  {integer-5  }                             {integer-6  }

        [, LINES AT TOP  {data-name-6}]  [, LINES AT BOTTOM  {data-name-7}]]
                         {integer-7  }                       {integer-8  }

    [; CODE-SET IS alphabet-name].

[record-description-entry] ... ] ... ]

[WORKING-STORAGE SECTION.

[77-level-description-entry ]
[record-description-entry   ]  ...]

[LINKAGE SECTION.

[77-level-description-entry ]
[record-description-entry   ]  ...]

[GLOBAL SECTION.

[77-level-description-entry ]
[record-description-entry   ]  ...]
```


TEXT EDITOR ERROR MESSAGES

FILE CANNOT BE ACCESSED — File-id assigned to file having an invalid characteristic for either CONCAT or SAVE operation.

FILE NOT FOUND — EWF or file-id not previously assigned.

ILLEGAL COMMAND — Invalid text editor command code or structure.

INSUFFICIENT SPACE IN FILE — EWF or file-id cannot accommodate number of records to be written.

ITEM NOT FOUND — Sequence number not present in EWF.
- Relative record number beyond number of records on EWF.
- POSITION command would position beyond first or last record in EWF.
- SEARCH character-string not present in EWF.

MOVE FROM RECORD NOT FOUND — First record in MOVE command not found in EWF.

MOVE TO RECORD NOT FOUND — Second record in MOVE command not found in EWF.

WARNING: SEQ NUMBER OVERLAP — MOVE or RESEQUENCE command causes sequence number to exceed that of record following range to be sequenced.

WORK FILE IS EMPTY — No data records in EWF.

WORK FILE MUST BE OWNED — When EWF is nonprivate, its ASSIGN system command must have the OWN parameter.

RUNTIME ERROR MESSAGES

COBOL ERROR (BINARY OPTION REQUIRED) — Binary runtime module is needed.

COBOL ERROR (BOOLEAN OPTION REQUIRED) — Boolean runtime module is needed.

COBOL ERROR (CRT I/O OPTION REQUIRED) — CRT input-output runtime module is needed.

COBOL ERROR (COMPILATION) — Error in COBOL statement being executed.

COBOL ERROR (INVALID DOPE) — Illegal object construction.

COBOL ERROR (INVALID EDIT) — Illegal object construction.

COBOL ERROR (LINAGE OPTION REQUIRED) — Linage runtime module is needed.

COBOL ERROR (MUL/DIV OPTION REQUIRED) — Multiply/divide runtime module is needed.

COBOL ERROR (NUMBER OF PARAMETERS) — Number of data-names in associated USING phrases are not equal.

COBOL ERROR (OUT OF MEMORY) — Insufficient memory for called program.

COBOL ERROR (PARAMETER LENGTH ERROR) — Data item in called program does not have sufficient memory space allocated by the calling program.

COBOL ERROR (STRING OPTION REQUIRED) — The string runtime module is needed.

COBOL ERROR (SUBSCRIPT RANGE) — Value of subscript or index is invalid.

COBOL ERROR (UNDEFINED LABEL REF) — Statement being executed refers to an undefined procedure-name.

COMPILER DIAGNOSTIC MESSAGES

RECORD KEY INVALID – Record key data item not alphanumeric item within record description of indexed file.
RECORD KEY NOT ALPHANUMERIC – Record key data item not alphanumeric.
RECORD KEY NOT IN A RECORD – Record key data item not within record description of indexed file.
RECORD KEY OUT OF MINIMUM LIMIT – Record key data item larger than minimum length record of indexed file.
RECORD KEY UNDEFINED – No data description entry for record key data item.
REDEFINES – Redefinition error.
REFERENCE TYPE – Data type of identifier invalid in context.
RELATION – Incompatible operands in relation condition.
RELATIVE KEY – RELATIVE KEY clause invalid or missing for relative file.
RELATIVE KEY INVALID – Relative key data item not unsigned integer.
RELATIVE KEY UNDEFINED – No data description entry for relative key data item.
RELATIVE VARIABLE CLASH – Relative file must not contain variable length records.
RESERVED – Reserved word given where user-defined word required.
RESERVED WORD CONFLICT – Reserved word given where user-defined word required; but reserved word has no function in this compiler.
SCAN RESUME – Processing of source program resumed.
SECTION ILLEGAL – Section header illegal.
SEGMENT – Segment-number of independent segment not 50 through 99.
SEPARATOR – Redundant punctuation.
SIGN – SIGN clause conflicts with USAGE and PICTURE clauses.
SIZE – Size of data item incorrect in context.
– Size of user-defined word exceeds 30 characters.
SIZE ERROR – Size of specified record, data item, or literal is invalid.
SUBSCRIPT – Incorrect number of subscripts or indexes.
SYNC – SYNCHRONIZED clause must not be given for a group data item.
SYNTAX – Incorrect character or reserved word for context.
– Incorrect positioning in columns 8 through 11.
– Unexpected character in column 7.
TOO MANY CLAUSES – Too many AFTER phrases in PERFORM VARYING statement.
TOP INVALID – Invalid value in LINES AT TOP phrase of LINAGE clause.
TOP UNDEFINED – No data description entry for data item in LINES AT TOP phrase of LINAGE clause.
UNDEFINED – File-name in file description entry is not defined in file control entry.
– Procedure-name referenced in declaratives not defined within declaratives.
UNDEFINED FILE – File does not have file description entry.
UNDEFINED PROCEDURE – Procedure-name not assigned to a paragraph or section.
USE PROCEDURE OR CONDITION REQUIRED – Input-output statement requires INVALID KEY, AT END, or associated USE procedure.
USED AS DATA-NAME – Procedure-name used as data-name.
VALUE – VALUE IS clause conflicts with other clauses in entry.
– Value of literal is incorrect in context.
VALUE SIZE – VALUE IS clause has value exceeding size of PICTURE character-string.

GENERAL FORMAT FOR DATA DESCRIPTION ENTRY

```
level-number  { data-name-1 }
              { FILLER      }

  [; REDEFINES data-name-2]

  [; { PICTURE } IS character-string]
     { PIC     }

  [; [USAGE IS] { BIT                  }]
                { COMPUTATIONAL        }
                { COMP                 }
                { COMPUTATIONAL-3      }
                { COMP-3               }
                { COMPUTATIONAL-5      }
                { COMP-5               }
                { COMPUTATIONAL-6      }
                { COMP-6               }
                { DISPLAY              }
                { INDEX                }

  [; [SIGN IS] TRAILING [SEPARATE CHARACTER]]

  [; OCCURS integer TIMES [INDEXED BY index-name-1 [, index-name-2] ... ]]

  [; { SYNCHRONIZED } [ LEFT  ]]
     { SYNC         } [ RIGHT ]

  [; { JUSTIFIED } RIGHT]
     { JUST      }

  [; BLANK WHEN ZERO]

  [; VALUE IS literal].
```

GENERAL FORMAT FOR PROCEDURE DIVISION

FORMAT 1:

```
PROCEDURE DIVISION  [USING data-name-1  [, data-name-2] ... ].

[DECLARATIVES.

{section-name SECTION  [segment-number].  declarative-sentence

[paragraph-name.  [sentence] ... ] ... } ...

END DECLARATIVES.]

{section-name SECTION  [segment-number].

[paragraph-name.  [sentence] ... ] ... } ...
```

FORMAT 2:

```
PROCEDURE DIVISION  [USING data-name-1  [, data-name-2] ... ].

{paragraph-name.  [sentence] ... } ...
```

FORMAT FOR COPY STATEMENT

```
COPY text-name [, sequence-number-1 {THROUGH | THRU} {sequence-number-2 | END}].
```

FORMAT FOR USE STATEMENT (A DECLARATIVE SENTENCE)

```
USE AFTER STANDARD {EXCEPTION | ERROR} PROCEDURE ON {file-name-1 [, file-name-2] ... | INPUT | OUTPUT | I-O | EXTEND}.
```


COMPILER DIAGNOSTIC MESSAGES

KEY REQUIRED – RECORD KEY clause missing from file control entry of indexed file.
– RELATIVE KEY clause missing from file control entry of relative file.

LABEL – LABEL RECORDS clause conflicts with device type or is missing.

LEVEL – Invalid level-number.

LITERAL VALUE – Value of literal incorrect in context.

MAXIMUM RECORD SIZE LIMIT – Bytes in record description entry exceed bytes in RECORD clause.

MINIMUM RECORD SIZE LIMIT – Bytes in record description entry less than bytes in RECORD clause.

MOVE – Operands of MOVE statement specify an invalid move operation.

MUST BE INTEGER – Context requires a decimal integer.

MUST BE LINKAGE SECTION – Data-name in USING phrase of Procedure Division header is not in Linkage Section.

MUST BE PROCEDURE – Context requires a procedure-name.

MUST BE SECTION – Context requires procedure-name to be section-name.

MUST BE 01 OR 77 LEVEL DATA ITEM – Data-name in USING phrase of Procedure Division header is neither an 01 nor 77 level entry.

MUST HAVE INDEX-NAME – Identifier in SEARCH must have an INDEXED BY phrase.

NESTING – Illegal nesting of condition that is not an IF condition.

NOT IN REDEFINES – OCCURS or VALUE clause invalid in redefining data item.

NOT IN THIS SECTION – VALUE IS clause illegal in File or Linkage Section.

NOT IN USING – Linkage data item not referenced in USING phrase of Procedure Division header.

NOT WITH LINAGE – LINAGE clause and ADVANCING mnemonic-name associated with same file.

NOT WITH THIS OBJECT-COMPUTER – COBOL element not supported by specified object computer.

OCCURS – OCCURS clause incorrectly specified in 01 or 77 entry.
– OCCURS clause defining table of more than 3 dimensions.

OCCURS-VALUE CLASH – VALUE and OCCURS clause must not be specified in same entry.

PERIOD REQUIRED – Period is required.

PICTURE – Invalid PICTURE syntax.

PICTURE-USAGE CLASH – USAGE clause conflicts with usage implied by PICTURE.

PROCEDURE INDEPENDENCE – PERFORM in independent segment incorrectly refers to another independent segment.

QUALIFICATION – Insufficient or incorrect qualification.

RECORD DEPENDING DATA ITEM ILLEGALLY DEFINED – Data item in DEPENDING ON phrase of RECORD IS VARYING clause must be in Working-Storage or Linkage Sections.

RECORD DEPENDING DATA ITEM INVALID – Data item in DEPENDING ON phrase of RECORD IS VARYING clause not elementary unsigned integer.

RECORD DEPENDING DATA ITEM TOO SMALL – Data item in DEPENDING ON phrase of RECORD IS VARYING clause smaller than maximum length record.

RECORD DEPENDING DATA ITEM UNDEFINED – No data description entry for data item in DEPENDING ON phrase of RECORD IS VARYING clause.

RECORD KEY – RECORD KEY clause invalid or missing for indexed file.

COMPILER DIAGNOSTIC MESSAGES

ACCESS CLASH – Incorrect access is given for the specified file organization.

ASSIGN – ASSIGN clause not given in file control entry.

BLANK WHEN ZERO – BLANK WHEN ZERO clause given for numeric or group item.

BLOCK CONFLICT – Number of characters in BLOCK CONTAINS clause not multiple of record size in RECORD clause.

BLOCK SIZE EXCEEDS DEVICE SIZE – Number of characters in BLOCK CONTAINS clause exceeds largest block size peripheral accepts.

BOTTOM INVALID – Invalid value in LINES AT BOTTOM phrase.

BOTTOM UNDEFINED – No data description entry for data item in LINES AT BOTTOM phrase of LINAGE clause.

COMPOSITE OVERFLOW – Composite of operands exceeds 18 digits.

COMPUTER – Invalid computer-name.

COPY – Incorrect COPY statement

DEVICE CLASH – Device other than disc specified for relative or indexed file.

DEVICE TYPE – File characteristic or operation inconsistent for device type.

DOUBLE DECLARATION – Same clause appears twice in one entry.

DOUBLE DEFINITION – Multiple entries for same data-name within one hierarchy.
– Multiple paragraphs have same paragraph-name within one section.
– Same section-name assigned to different sections.

DUPLICATE – Multiple USE procedures declared for same function.

DUPLICATE PARAMETER – Same data-name appears more than once in USING phrase of Procedure Division header.

EXTENSION – Redefined data item is larger or smaller than original data item.

FILE IDENTIFIER – VALUE OF FILE-ID clause appears twice in one entry.
– VALUE OF LABEL clause should be VALUE OF FILE-ID clause.

FILE NAME REQUIRED – File-name needed after verb or FD level indicator.

FILE STATUS INVALID – File status data item not two-character alphanumeric data item.

FILE STATUS UNDEFINED – No data description entry for file status data item.

FILE TYPE – DELETE or START invalid on sequential file.
– AT END phrase invalid on random access READ statement.

FOOTING INVALID – Invalid value in FOOTING phrase of LINAGE clause.

FOOTING UNDEFINED – No data description entry for data item in FOOTING phrase of LINAGE clause.

GROUP CLASH – USAGE or VALUE clause of item within group conflicts with group's USAGE or VALUE clause.

GROUP VALUE CLASH – VALUE clause specified for group item having subordinate items with differing data types.

IDENTIFIER – Undefined identifier.

ILLEGAL ALTER – ALTER statement refers to paragraph incorrectly structured.

ILLEGAL PERFORM – PERFORM statement refers to paragraph or section incorrectly segmented.

INVALID CHARACTER – Illegal character encountered.

INVALID FILE – Associated file description entry has an error.

JUSTIFY – JUSTIFIED clause conflicts with other clauses in entry.

GENERAL FORMAT FOR VERBS

```
ACCEPT identifier-1 [, LINE {identifier-2 | literal-1}] [, POSITION {identifier-3 | literal-2}]
  [, SIZE {identifier-4 | literal-3}] [, PROMPT] [, ECHO] [, CONVERT]

ACCEPT identifier FROM {DATE | DAY | TIME}

ADD {identifier-1 | literal-1} [, identifier-2 | , literal-2] ... TO identifier-m [ROUNDED]
  [; ON SIZE ERROR imperative-statement]

ADD {identifier-1 | literal-1} , {identifier-2 | literal-2} [, identifier-3 | , literal-3] ...
  GIVING identifier-m [ROUNDED] [; ON SIZE ERROR imperative-statement]

ALTER procedure-name-1 TO [PROCEED TO] procedure-name-2
  [, procedure-name-3 TO [PROCEED TO] procedure-name-4] ...

CALL {identifier-1 | literal-1} [USING data-name-1 [, data-name-2] ... ]
  [; ON OVERFLOW imperative-statement]

CANCEL {identifier-1 | literal-1} [, identifier-2 | , literal-2] ...

CLOSE file-name-1 [ {REEL | UNIT} [WITH NO REWIND] | WITH {NO REWIND | LOCK} ]
  [ , file-name-2 [ {REEL | UNIT} [WITH NO REWIND] | WITH {NO REWIND | LOCK} ] ] ...

COMPUTE identifier = boolean-expression

DELETE file-name RECORD

DELETE file-name RECORD [; INVALID KEY imperative-statement]
```

GENERAL FORMAT FOR VERBS

```
DISPLAY {identifier-1} [, LINE {identifier-2}] [, POSITION {identifier-3}]
        {literal-1    }        {literal-2    }            {literal-3    }
        [, SIZE {identifier-4}] [, ERASE]
                {literal-4    }
  [{, identifier-5} [, LINE {identifier-6}] [, POSITION {identifier-7}]
  {, literal-5    }        {literal-6    }            {literal-7    }
        [, SIZE {identifier-8}] [, ERASE] ]...
                {literal-8    }
```

```
DIVIDE {identifier-1} INTO identifier-2 [ROUNDED]
       {literal-1    }
  [; ON SIZE ERROR imperative-statement]
```

```
DIVIDE {identifier-1} INTO {identifier-2} GIVING identifier-3 [ROUNDED]
       {literal-1    }     {literal-2    }
  [REMAINDER identifier-4] [; ON SIZE ERROR imperative-statement]
```

```
DIVIDE {identifier-1} BY {identifier-2} GIVING identifier-3 [ROUNDED]
       {literal-1    }    {literal-2    }
  [REMAINDER identifier-4] [; ON SIZE ERROR imperative-statement]
```

```
EXIT.
```

```
EXIT PROGRAM.
```

```
GO TO procedure-name
```

```
GO TO procedure-name-1 [, procedure-name-2] ... , procedure-name-n
   DEPENDING ON identifier
```

```
IF condition; { statement-1; ELSE statement-2 }
              { NEXT SENTENCE; ELSE statement-2 }
              { statement-1; ELSE NEXT SENTENCE }
              { statement-1.                    }
```

```
INSPECT identifier-1 TALLYING
   identifier-2 FOR { {ALL     } {identifier-3} } [{BEFORE} INITIAL {identifier-4}]
                    { {LEADING} {literal-1    } }  {AFTER }         {literal-2    }
                    { CHARACTERS                }
```


NCR IMOS COBOL FILES

File Media	ASSIGN TO	File Organization	Access Mode
Punched card file	CARD-READER	Sequential	Sequential access
Printer file	PRINTER	Sequential	Sequential access
Magnetic tape file on a cassette	CASSETTE	Sequential	Sequential access
Magnetic tape file on a reel	MAGNETIC-TAPE	Sequential	Sequential access
Magnetic disc file	DISC	Sequential	Sequential access
		Relative	Sequential access Random access Dynamic access
		Indexed	Sequential access Random access Dynamic access

PERMISSIBLE INPUT-OUTPUT STATEMENTS

File Access Mode	Statement \ Open Mode	File Organization: Sequential File Input	Sequential File Output	Sequential File I-O	Sequential File Extend	Relative File Input	Relative File Output	Relative File I-O	Relative File Extend	Indexed File Input	Indexed File Output	Indexed File I-O	Indexed File Extend
Sequential	READ	X		X		X		X		X		X	
	READ NEXT												
	WRITE		X		X		X				X		
	REWRITE			X				X				X	
	START					X		X		X		X	
	DELETE							X				X	
Random	READ					X		X		X		X	
	READ NEXT												
	WRITE						X	X			X	X	
	REWRITE							X				X	
	START												
	DELETE							X				X	
Dynamic	READ					X		X		X		X	
	READ NEXT					X		X		X		X	
	WRITE						X	X			X	X	
	REWRITE							X				X	
	START					X		X		X		X	
	DELETE							X				X	

NCR IMOS III CONTROL STRINGS

Execute COBOL Object Program

```
ASSIGN input-disc-file-id TO physical-file-id(n),NOAUTO,READ,BUFFER,UPDATE
ASSIGN output-disc-file-id TO physical-file-id(n),NEW,s,AP,NOAUTO,BUFFER,UPDATE
ASSIGN card-file-id TO (n,CR)
ASSIGN mt-cassette-file-id TO (n,CA)
ASSIGN mt-reel-file-id TO (n,MT)
ASSIGN printer-file-id TO (n,LP)
ASSIGN printer-file-id TO spool-physical-file-id(n),NEW,s,AP,SPOOL
EXECUTE progname99(n),DEBUG
```

Create Control String on Magnetic Disc

```
ASSIGN EWF TO EWFn(n)
ASSIGN B TO cntlstrg(n),NEW,s,AP
EXECUTE $EDIT
   (system commands and text editor commands)
\
SAVE B
QUIT
```

Create Control String on Magnetic Tape

```
ASSIGN EWF TO EWFn(n)
ASSIGN B TO cntlstrg(n,CA)
EXECUTE $EDIT
   (system commands and text editor commands)
\
SAVE B
QUIT
```

Execute Control String on Magnetic Disc

```
EXECUTE cntlstrg(n),NOLIST
```

Execute Control String on Magnetic Tape

```
EXECUTE cntlstrg(n,CA),NOLIST
```

Execute Control String on Punched Cards

```
EXECUTE cntlstrg(n,CR),NOLIST
```


GENERAL FORMAT FOR VERBS

```
INSPECT identifier-1 REPLACING

  { { ALL      } { identifier-5 } }
  { { LEADING  } { literal-3    } }  BY { identifier-6 } [ { BEFORE } INITIAL { identifier-7 } ]
  { { FIRST    }                  }      { literal-4    }  [ { AFTER  }        { literal-5    } ]
  { CHARACTERS                    }

INSPECT identifier-1 TALLYING

                  { { ALL      } { identifier-3 } }
  identifier-2 FOR { { LEADING  } { literal-1    } } [ { BEFORE } INITIAL { identifier-4 } ]
                  { CHARACTERS                   }  [ { AFTER  }        { literal-2    } ]

  REPLACING

  { { ALL      } { identifier-5 } }
  { { LEADING  } { literal-3    } }  BY { identifier-6 } [ { BEFORE } INITIAL { identifier-7 } ]
  { { FIRST    }                  }      { literal-4    }  [ { AFTER  }        { literal-5    } ]
  { CHARACTERS                    }

MOVE { identifier-1 } TO identifier-2 [, identifier-3] ...
     { literal      }

MOVE { identifier } TO JCL-CODE
     { literal    }

MULTIPLY { identifier-1 } BY identifier-2 [ROUNDED]
         { literal-1    }

  [; ON SIZE ERROR imperative-statement]

MULTIPLY { identifier-1 } BY { identifier-2 } GIVING identifier-3 [ROUNDED]
         { literal-1    }    { literal-2    }

  [; ON SIZE ERROR imperative-statement]

      { INPUT file-name-1 [WITH NO REWIND] [, file-name-2 [WITH NO REWIND]] ... }
OPEN  { OUTPUT file-name-3 [WITH NO REWIND] [, file-name-4 [WITH NO REWIND]] ... }  ...
      { I-O file-name-5 [, file-name-6] ...                                      }
      { EXTEND file-name-7 [, file-name-8] ...                                   }

PERFORM procedure-name-1 [ { THROUGH } procedure-name-2 ]
                           { THRU    }

PERFORM procedure-name-1 [ { THROUGH } procedure-name-2 ] { identifier } TIMES
                           { THRU    }                    { integer    }

PERFORM procedure-name-1 [ { THROUGH } procedure-name-2 ] UNTIL condition-1
                           { THRU    }
```

GENERAL FORMAT FOR VERBS

```
PERFORM procedure-name-1 [ {THROUGH} procedure-name-2 ]
                         {THRU   }

    VARYING {identifier-1} FROM {identifier-2} BY {identifier-3}
            {index-name-1}      {index-name-2}    {literal-2   }
                                {literal-1   }

        UNTIL condition-1

PERFORM procedure-name-1 [ {THROUGH} procedure-name-2 ]
                         {THRU   }

    VARYING {identifier-1} FROM {identifier-2} BY {identifier-3}
            {index-name-1}      {index-name-2}    {literal-2   }
                                {literal-1   }

        UNTIL condition-1

        AFTER {identifier-4} FROM {identifier-5} BY {identifier-6}
              {index-name-3}      {index-name-4}    {literal-4   }
                                  {literal-3   }

            UNTIL condition-2

PERFORM procedure-name-1 [ {THROUGH} procedure-name-2 ]
                         {THRU   }

    VARYING {identifier-1} FROM {identifier-2} BY {identifier-3}
            {index-name-1}      {index-name-2}    {literal-2   }
                                {literal-1   }

        UNTIL condition-1

        AFTER {identifier-4} FROM {identifier-5} BY {identifier-6}
              {index-name-3}      {index-name-4}    {literal-4   }
                                  {literal-3   }

            UNTIL condition-2

            AFTER {identifier-7} FROM {identifier-8} BY {identifier-9}
                  {index-name-5}      {index-name-6}    {literal-6   }
                                      {literal-5   }

                UNTIL condition-3

READ file-name RECORD [INTO identifier] [; AT END imperative-statement]

READ file-name RECORD [INTO identifier] [; INVALID KEY imperative-statement]

READ file-name NEXT RECORD [INTO identifier] [; AT END imperative-statement]

REWRITE record-name [FROM identifier]

REWRITE record-name [FROM identifier] [; INVALID KEY imperative-statement]
```


NCR IMOS III CONTROL STRINGS

Print Editor Work File

```
ASSIGN A TO prognameO1(n)
ASSIGN EWF TO EWFn(n)
ASSIGN LO TO (n,LP)
EXECUTE $EDIT
CONCAT A
PRINT *1,10000
QUIT
```

Execute COBOL Compiler Program With Card Input

```
ASSIGN SI TO (n,CR)
ASSIGN BO TO progname99(n),NEW,s,AP
ASSIGN LO TO (n,LP)
EXECUTE $COBOL
```

Execute COBOL Compiler Program with Magnetic Disc Input

```
ASSIGN SI TO prognameO1(n)
ASSIGN BO TO progname99(n),NEW,s,AP
ASSIGN LO TO (n,LP)
EXECUTE $COBOL
```

Execute COBOL Compiler Program with Magnetic Tape Input

```
ASSIGN SI TO prognameO1(n,CA)
ASSIGN BO TO progname99(n),NEW,s,AP
ASSIGN LO TO (n,LP)
EXECUTE $COBOL
```

Execute COBOL Compiler Program with Listing to Spool File

```
ASSIGN SI TO prognameO1(n)
ASSIGN BO TO progname99(n),NEW,s,AP
ASSIGN LO TO prognameSP(n),NEW,s,AP,SPOOL
EXECUTE $COBOL
```

Print Spool File

```
ASSIGN A TO prognameSP(n)
ASSIGN B TO (n,LP)
MOVE A TO B
```

Execute COBOL Compiler Program with COPY Statement in Source Program

```
ASSIGN SI TO prognameO1(n)
ASSIGN textname TO physical-file-id(n)
ASSIGN BO TO progname99(n),NEW,s,AP
ASSIGN LO TO (n,LP)
EXECUTE $COBOL
```

NCR IMOS III CONTROL STRINGS

Create Editor Work File

```
ASSIGN EWF TO EWFn(n),PRI,NEW,500
```

Recover Content of Editor Work File

```
ASSIGN EWF TO EWFn(n)
ASSIGN B TO progname(n),NEW,s,AP
EXECUTE $EDIT,RECOVER
SAVE B
QUIT
```

Punched Card Input to Text Editor Program

```
ASSIGN A TO (n,CR)
ASSIGN EWF TO EWFn(n)
ASSIGN B TO progname01(n),NEW,s,AP
EXECUTE $EDIT
CONCAT A
SAVE B
QUIT
```

CRT Input to Text Editor Program

```
ASSIGN EWF TO EWFn(n)
ASSIGN B TO progname01(n),NEW,s,AP
EXECUTE $EDIT
INSERT *
   (COBOL source program lines)
\
SAVE B
QUIT
```

Changes to Source Program File Creating New Version

```
ASSIGN A TO progname01(n)
ASSIGN EWF TO EWFn(n)
ASSIGN B TO progname02(n),NEW,s,AP
EXECUTE $EDIT
CONCAT A
   (changes to source program file)
SAVE B
QUIT
```

Changes to Source Program File Replacing Old Version

```
ASSIGN A TO progname01(n),OWN
ASSIGN EWF TO EWFn(n)
EXECUTE $EDIT
CONCAT A
   (changes to source program file)
SAVE A
QUIT
```


GENERAL FORMAT FOR VERBS

```
SEARCH identifier-1 [ VARYING { identifier-2 } ]
                              { index-name-1 }
  [; AT END imperative-statement-1]
   ; WHEN condition-1 { imperative-statement-2 }
                      { NEXT SENTENCE          }
  [; WHEN condition-2 { imperative-statement-3 } ] ...
                      { NEXT SENTENCE          }

SET index-name-1 [, index-name-2] ... TO { index-name-3      }
                                         { index-data-item   }
                                         { identifier-1      }
                                         { integer-1         }

SET index-name-1 [, index-name-2] ... { UP BY   } { identifier-2 }
                                      { DOWN BY } { integer-2    }

SET index-data-item-1 [, index-data-item-2] ... TO { index-name      }
                                                   { index-data-item-3 }

SET identifier-1 [, identifier-2] ... TO index-name

SHIFT identifier-1 { RIGHT } { identifier-2 } [ POSITION  ]
                   { LEFT  } { integer-1    } [ POSITIONS ]

START file-name [ KEY { IS EQUAL TO          } data-name ]
                      { IS =                 }
                      { IS GREATER THAN      }
                      { IS >                 }
                      { IS NOT LESS THAN     }
                      { IS NOT <             }
  [; INVALID KEY imperative-statement]

STOP { RUN     }
     { literal }

STRING { identifier-1 } [, identifier-2 ] ... DELIMITED BY { identifier-3 }
       { literal-1    }  [, literal-2    ]                 { literal-3    }
                                                           { SIZE         }
  [, { identifier-4 } [, identifier-5 ] ... DELIMITED BY { identifier-6 } ] ...
     { literal-4    }  [, literal-5    ]                 { literal-6    }
                                                         { SIZE         }
   INTO identifier-7 [WITH POINTER identifier-8]
  [; ON OVERFLOW imperative-statement]

SUBTRACT { identifier-1 } [, identifier-2 ] ... FROM identifier-m [ROUNDED]
         { literal-1    }  [, literal-2    ]
  [; ON SIZE ERROR imperative-statement]
```

GENERAL FORMAT FOR VERBS

```
SUBTRACT {identifier-1} [, identifier-2] ... FROM {identifier-m}
         {literal-1   } [, literal-2   ]      {literal-m   }
  GIVING identifier-n [ROUNDED] [; ON SIZE ERROR imperative-statement]

UNSTRING identifier-1
  [DELIMITED BY [ALL] {identifier-2} [, OR [ALL] {identifier-3}] ...]
                      {literal-1   }             {literal-2   }
    INTO identifier-4 [, DELIMITER IN identifier-5] [, COUNT IN identifier-6]
      [, identifier-7 [, DELIMITER IN identifier-8] [, COUNT IN identifier-9]] ...
    [WITH POINTER identifier-10] [TALLYING IN identifier-11]
    [; ON OVERFLOW imperative-statement]

WRITE record-name [FROM identifier-1]
  [{BEFORE} ADVANCING {{identifier-2} [LINE ]}]
   {AFTER }           {{integer     }  [LINES]}
                      {{mnemonic-name}       }
                      {{PAGE        }       }
  [; AT {END-OF-PAGE} imperative-statement]
        {EOP        }

WRITE record-name [FROM identifier] [; INVALID KEY imperative-statement]
```


GENERAL FORMAT FOR NCR IMOS III TEXT EDITOR COMMANDS

String

```
STRING 'character-string-1' WITH 'character-string-2'
STRING 'character-string-1','character-string-2'
STRING 'character-string-1' WITH 'character-string-2',ALL
STRING 'character-string-1','character-string-2',ALL
STRING 'character-string-1' WITH 'character-string-2',n
STRING 'character-string-1','character-string-2',n
```

GENERAL FORMAT FOR NCR IMOS III TEXT EDITOR COMMANDS

Move Source Line Specified by Relative Record Number

```
MOVE DELETE,*n-p
MOVE DELETE,*m,n-p
```

Position Current Source Line + n

```
POSITION n
```

Position Current Source Line – n

```
POSITION -n
```

Print Source Line Specified by Sequence Number

```
PRINT x
PRINT x,y
```

Print Source Line Specified by Relative Record Number

```
PRINT *n
PRINT *m,n
```

Quit Execution of Text Editor Program

```
QUIT
```

Resequence

```
RESEQUENCE z,i
RESEQUENCE x,y-z,i
```

Save Variable Length Record File

```
SAVE file-id
SAVE file-id(,b)
```

Save Fixed Length Record File

```
SAVE file-id(r,b)
SAVE file-id(r)
```

Search

```
SEARCH 'string'
SEARCH 'string',n
```


GENERAL FORMAT FOR CONDITIONS

RELATION CONDITION:

```
{identifier-1}   {IS [NOT] GREATER THAN}   {identifier-2}
{literal-1   }   {IS [NOT] LESS THAN   }   {literal-2   }
{index-name-1}   {IS [NOT] EQUAL TO    }   {index-name-2}
                 {IS [NOT] >           }
                 {IS [NOT] <           }
                 {IS [NOT] =           }
```

CLASS CONDITION:

```
identifier IS [NOT] {NUMERIC   }
                    {ALPHABETIC}
```

SIGN CONDITION:

```
identifier IS [NOT] {POSITIVE}
                    {NEGATIVE}
                    {ZERO    }
```

SWITCH-STATUS CONDITION:

```
condition-name
```

COMBINED CONDITION:

```
condition-1 {{AND}} condition-2 } ...
            {{OR }}
```

MISCELLANEOUS FORMATS

QUALIFICATION:

```
{record-name}  [{OF}  file-name]
{data-name}    {IN}

data-name-1 [{OF}  data-name-2] ...
            {IN}

paragraph-name [{OF}  section-name]
               {IN}
```

SUBSCRIPTING:

```
data-name (subscript-1 [, subscript-2 [, subscript-3]])
```

INDEXING:

```
data-name ( {index-name-1 [{±} literal-2]}
            {literal-1                  }
  [, {index-name-2 [{±} literal-4]} [, {index-name-3 [{±} literal-6]}]])
     {literal-3                   }    {literal-5                   }
```

IDENTIFIER: FORMAT 1

```
data-name-1 [{OF}  data-name-2] ...
            {IN}
  [(subscript-1 [, subscript-2 [, subscript-3]])]
```

IDENTIFIER: FORMAT 2

```
data-name-1 [{OF}  data-name-2] ... [( {index-name-1 [{±} literal-2]}
            {IN}                       {literal-1                  }
  [, {index-name-2 [{±} literal-4]} [, {index-name-3 [{±} literal-6]}]])]
     {literal-3                   }    {literal-5                   }
```


GENERAL FORMAT FOR NCR IMOS III TEXT EDITOR COMMANDS

Delete Source Line Specified by Sequence Number

```
DELETE x
DELETE x,y
```

Delete Source Line Specified by Relative Record Number

```
DELETE *n
DELETE *m,n
```

Display Current Source Line

```
DISPLAY
```

Display Source Line Specified by Sequence Number

```
DISPLAY x
DISPLAY x,y
```

Display Source Line Specified by Relative Record Number

```
DISPLAY *n
DISPLAY *m,n
```

Find Source Line Specified by Sequence Number

```
FIND x
```

Find Source Line Specified by Relative Record Number

```
FIND *n
```

Insert 80-Character Source Line

```
INSERT *D
```

Insert 72-Character Source Line

```
INSERT D
```

Move Source Line Specified by Sequence Number

```
MOVE DELETE,x-z,i
MOVE DELETE,x,y-z,i
```

GENERAL FORMAT FOR NCR IMOS III TEXT EDITOR COMMANDS

Change Current Source Line

```
CHANGE
source-line-1                    replacement variation

CHANGE
new line key                     no change variation

CHANGE
backslash (\) key                no change variation
```

Change Source Line Specified by Sequence Number

```
CHANGE x
source-line-1                    replacement variation

CHANGE x
new line key                     no change variation

CHANGE x
backslash (\) key                no change variation
```

Change Source Line Specified by Relative Record Number

```
CHANGE *n
source-line-1                    replacement variation

CHANGE *n
new line key                     no change variation

CHANGE *n
backslash (\) key                no change variation
```

Concat Entire File

```
CONCAT file-id(r,b)
```

Concat Records Specified by Sequence Number

```
CONCAT file-id(r,b),x

CONCAT file-id(r,b),x,y
```

Concat Records Specified by Relative Record Number

```
CONCAT file-id(r,b),*n

CONCAT file-id(r,b),*,n

CONCAT file-id(r,b),*m,n
```

Delete Current Source Line

```
DELETE
```


FIGURATIVE CONSTANTS

ZERO	HIGH-VALUE
ZEROS	HIGH-VALUES
ZEROES	LOW-VALUE
SPACE	LOW-VALUES
SPACES	QUOTE
	QUOTES

PICTURE CHARACTERS

Data Characters

A	alphabetic character or a space
X	any character from ASCII character set
9	numbers 0 through 9
1	boolean characters 0 and 1

Special Usage Characters

S	Sign
V	Assumed decimal point location
P	Assumed decimal scaling position

Suppression and Replacement Characters

Z	Leading zeros replaced by spaces
*	Leading zeroes replaced by asterisks

Insertion Characters

cs	Currency symbol such as $; fixed or floating
,	Comma
.	Period; the actual decimal point
B	Blank
0	Zero
/	Stroke

Editing Sign Control Symbols

–	Minus sign; fixed left or right, floating on left
+	Plus sign; fixed left or right, floating on left
CR	Credit symbol; right
DB	Debit symbol; right

Editing Symbol	Data Item Positive or Zero	Data Item Negative
–	ø	–
+	+	–
CR	øø	CR
DB	øø	DB

NCR IMOS COBOL RESERVED WORD LIST

ACCEPT
ACCESS
ADD
ADVANCING
AFTER
ALL
ALPHABETIC
ALSO
ALTER
ALTERNATE
AND
ARE
AREA
AREAS
ASCENDING
ASSIGN
AT
AUTHOR

BEFORE
BIT
BLANK
BLOCK
BOTTOM
BY

CALL
CANCEL
CD
CF
CH
CHARACTER
CHARACTERS
CLOCK-UNITS
CLOSE
COBOL
CODE
CODE-SET
COLLATING
COLUMN
COMMA
COMMUNICATION
COMP
COMP-3
COMP-5
COMP-6
COMPUTATIONAL
COMPUTATIONAL-3
COMPUTATIONAL-5
COMPUTATIONAL-6
COMPUTE
CONFIGURATION
CONTAINS
CONTROL
CONTROLS
CONVERT
COPY
CORR
CORRESPONDING
COUNT
CURRENCY

DATA
DATE
DATE-COMPILED
DATE-WRITTEN
DAY
DE
DEBUG-CONTENTS
DEBUG-ITEM
DEBUG-LINE
DEBUG-NAME
DEBUG-SUB-1
DEBUG-SUB-2
DEBUG-SUB-3
DEBUGGING
DECIMAL-POINT
DECLARATIVES
DELETE
DELIMITED
DELIMITER
DEPENDING
DESCENDING
DESTINATION
DETAIL
DISABLE
DISPLAY
DIVIDE
DIVISION
DOWN
DUPLICATES
DYNAMIC

ECHO
EGI
ELSE
EMI
ENABLE
END
END-OF-PAGE
ENTER
ENVIRONMENT
EOP
EQUAL
ERASE
ERROR
ESI
EVERY
EXCEPTION
EXIT
EXTEND

FD
FILE
FILE-CONTROL
FILLER
FINAL
FIRST
FOOTING
FOR
FROM

GENERATE
GIVING

GLOBAL
GO
GREATER
GROUP

HEADING
HIGH-VALUE
HIGH-VALUES

I-O
I-O-CONTROL
IDENTIFICATION
IF
IN
INCOMPLETE
INDEX
INDEXED
INDICATE
INITIAL
INITIATE
INPUT
INPUT-OUTPUT
INSPECT
INSTALLATION
INTO
INVALID
IS

JCL-CODE
JUST
JUSTIFIED

KEY

LABEL
LAST
LEADING
LEFT
LENGTH
LESS
LIMIT
LIMITS
LINAGE
LINAGE-COUNTER
LINE
LINE-COUNTER
LINES
LINKAGE
LOCK
LOW-VALUE
LOW-VALUES

MEMORY
MERGE
MESSAGE
MODE
MODULES
MOVE
MULTIPLE
MULTIPLY

NATIVE

GENERAL FORMAT FOR NCR IMOS III SYSTEM COMMANDS

Move a File

label:MOVE file-id-1 *TO* file-id-2,*PK*;*com*

Move Indexed File and Rebuild Key Index Directory

label:MOVE file-id-1 *TO* file-id-2,RB,k,n;*com*

Move All Disc Files

label:MOVE (n_1,*DI*) *TO* (n_2,*DI*);*com*

Remove Disc From System

label:REMOVE *volume-serial-number*(n,*DI*);*com*

Resume Activity of CRT Terminal

label:RESUME (n,*CT*);*com*

Set System Software Switch

label:SET SWITCH $n_1,n_2,n_3,n_4,n_5,n_6,n_7,n_8$ *TO* $\begin{Bmatrix}\text{ON}\\\text{OFF}\end{Bmatrix}$;*com*

Set JCL-CODE Data Item

label:SET JCL *TO* n;*com*

Set Mask of Unavailable System Commands

label:SET MASK (n,*CT*) *AS,AT,CH,DI,DU,MOU,MOV,SE,SU TO* $\begin{Bmatrix}\text{ON}\\\text{OFF}\end{Bmatrix}$;*com*

Set Peripheral Status

label:SET STATUS (n, $\begin{bmatrix}DI\\CA\\MT\\CR\\LP\end{bmatrix}$) *TO* $\begin{Bmatrix}\text{UP}\\\text{DOWN}\end{Bmatrix}$;*com*

Suspend Activity of CRT Terminal

label:SUSPEND (n,*CT*);*com*

GENERAL FORMAT FOR NCR IMOS III SYSTEM COMMANDS

Execute Object Program

label: EXECUTE physical-file-id(n,DI),DEBUG;com

Execute Control String on Magnetic Disc

label: EXECUTE physical-file-id(n,DI),NOLIST;com

Execute Control String on Magnetic Tape

label: EXECUTE physical-file-id(n,CA),NOLIST;com

Execute Control String on Punched Cards

label: EXECUTE (n,CR),NOLIST;com

Execute COBOL Compiler Program

label: EXECUTE $COBOL [,w / ,wK],DEBUG,LIST;com

Execute Text Editor Program

label: EXECUTE $EDIT,RECOVER;com

If JCL-CODE Data Item Equal

label-1: IF $n_1,n_2,n_3,n_4,n_5,n_6,n_7,n_8$ GO TO label-2;com

If JCL-CODE Data Item Less Than

label-1: IF < n_1 GO TO label-2;com

If JCL-CODE Data Item Greater Than

label-1: IF > n_1 GO TO label-2;com

If JCL-CODE Data Item Not Equal

label-1: IF < > n_1 GO TO label-2;com

Mount Disc Into System

label: MOUNT volume-serial-number(n,DI);com

NCR IMOS COBOL RESERVED WORD LIST

NEGATIVE
NEXT
NO
NOT
NUMBER
NUMERIC

OBJECT-COMPUTER
OCCURS
OF
OFF
OMITTED
ON
OPEN
OPTIONAL
OR
ORGANIZATION
OUTPUT
OVERFLOW
OVERLAP

PAGE
PAGE-COUNTER
PERFORM
PF
PH
PIC
PICTURE
PLUS
POINTER
POSITION
POSITIONS
POSITIVE
PRINTING
PROCEDURE
PROCEDURES
PROCEED
PROGRAM
PROGRAM-ID
PROMPT

QUEUE
QUOTE
QUOTES

RANDOM
RD
READ
RECEIVE
RECORD
RECORDS
REDEFINES
REEL
REFERENCES
RELATIVE
RELEASE
REMAINDER
REMOVAL
RENAMES
REPLACING
REPORT
REPORTING
REPORTS
RERUN
RESERVE
RESET
RETURN
REVERSED
REWIND
REWRITE
RF
RH
RIGHT
ROUNDED
RUN

SAME
SD
SEARCH
SECTION
SECURITY
SEGMENT
SEGMENT-LIMIT
SELECT
SEND
SENTENCE
SEPARATE
SEQUENCE
SEQUENTIAL
SET
SHIFT
SIGN
SIZE
SORT
SORT-MERGE
SOURCE
SOURCE-COMPUTER
SPACE
SPACES
SPECIAL-NAMES
STANDARD
STANDARD-1
START
STATUS
STOP
STRING
SUB-QUEUE-1
SUB-QUEUE-2
SUB-QUEUE-3
SUBTRACT
SUM
SUPPRESS
SYMBOLIC
SYNC
SYNCHRONIZED

TABLE
TALLYING
TAPE
TERMINAL
TERMINATE
TEST
TEXT
THAN
THROUGH
THRU
TIME
TIMES
TO
TOP
TRAILING
TRANSPORT
TYPE

UNIT
UNSTRING
UNTIL
UP
UPON
USAGE
USE
USING

VALUE
VALUES
VARYING

WHEN
WITH
WORDS
WORKING-STORAGE
WRITE

ZERO
ZEROES
ZEROS

\+
–
*
/
**
>
<
=

NCR IMOS COBOL SYSTEM-NAMES

BAR-CODE
CARD-READER
CASSETTE
CFF
DISC
FILE-ID
JOURNAL
LEDGER
LINE-n
MAGNETIC-TAPE
NCR-DPS
NCR-IMOS
NCR-IRX
PRINTER
SWITCH-n

GENERAL FORMAT FOR NCR IMOS III SYSTEM COMMANDS

Assign Input or I-O Magnetic Disc File

label:ASSIGN file-id TO physical-file-id/*gen*(n,DI) [,OWN / ,SHR],OLD,NOAUTO,READ,BUFFER,UPDATE;*com*

Assign Output Magnetic Disc File

label:ASSIGN file-id TO physical-file-id/*gen*(n,DI) [,OWN / ,SHR / ,PRI],NEW,s,AP [,SCRATCH / ,SPOOL],NOAUTO,BUFFER,UPDATE;*com*

Assign Magnetic Tape File

label:ASSIGN file-id TO physical-file-id(n,{CA / MT});*com*

Assign Punched Card File

label:ASSIGN file-id TO (n,CR);*com*

Assign Printer File

label:ASSIGN file-id TO (n,LP);*com*

Attach CRT Terminal

label:ATTACH (n,CT);*com*

Change Disc Physical File-Id

label:CHANGE physical-file-id-1/*gen*(n,DI) TO physical-file-id-2/*gen*;*com*

Delete Disc File

label:DELETE physical-file-id/*gen*(n,DI);*com*

Delete All Generations of a Disc File

label:DELETE physical-file-id/ALL(n,DI);*com*

Delete All Files From Disc Pack

label:DELETE volume-serial-number(n,DI);*com*

Detach CRT Terminal

Label:DETACH (n,CT);*com*

GENERAL FORMAT FOR NCR IMOS III SYSTEM COMMANDS

Display Disc Directory

label:DISPLAY volume-serial-number(n_1,DI) TO (n_2,LP);*com*

Display File Directory Entry

label:DISPLAY physical-file-id/*gen*(n_1,DI) TO (n_2,LP);*com*

Display System Status

label:DISPLAY STATUS TO (n,LP);*com*

Display System Software Switches

label:DISPLAY SWITCH TO (n,LP);*com*

Display JCL-CODE Data Item

label:DISPLAY JCL TO (n,LP);*com*

Display Mask of Unavailable System Commands

label:DISPLAY MASK TO (n,LP);*com*

Display CRT Message

label:DISPLAY 'message' TO (n_1,n_2,n_3, ... n_x);*com*

Dump Disc File

label:DUMP physical-file-id/*gen*(n_1,DI),b,e TO (n_2,LP);*com*

Dump Key Index Directory

label:DUMP physical-file-id/*gen*(n_1,DI),I TO (n_2,LP);*com*

Dump Memory

label:DUMP b,e TO (n,LP);*com*

label:DUMP &b,&e TO (n,LP);*com*

Dump Disc Sectors

label:DUMP &b,&e(n_1,DI) TO (n_2,LP);*com*